웹 클러스터 구성을 위한 시뮬레이션 분석

강성열† · 송영효††

Simulation Analysis for Configuring Web Clusters

Sungyeol Kang† · Younghyo Song††

ABSTRACT

High-volume web sites often use clusters of servers with load balancing as a way to increase the performance, scalability, and availability of the sites. Load balancing, usually performed by load balancer in front of such clusters, is a technique to spread workload between several computers or resources, in order to get optimal resource utilization or response time. In this paper we examine the performance for several configurations of cluster-based web servers using a simulation approach. We investigate two types of buffering scheme (common and local) for web clusters and three load balancing policies (uniformly random, round robin, and least queue first), using response time as a performance measure. We also examine two basic approaches of scaling web clusters: adding more servers of same type or upgrading the capacity of the servers in the clusters.

key words : web server, cluster, load balancing, buffering, response time

1. 서 론

대용량 웹 사이트들은 다수의 서버로 이루어진 클러스터를 흔히 사용한다. 그 클러스터 전단에 부하배분기(load balancer)를 두어서 그 여러 서버들에 요청되는 부하를 각 서버에 적절히 나누어 준다. 이러한 클러스터 구조를 사용함으로써 웹 사이트의 가용도(availability)와 확장성(scalability)을 용이하게 제고할 수 있다.

웹 클러스터의 용량을 확장하는 기본적인 접근으로, 같은 유형의 서버들을 증설하는 (scaling out, 수평적) 방식과 그 클러스터에 서버들의 용량을 업그레이드 하여 증설하는 (scaling up, 수직적) 방식을 들 수 있다. 이러한 웹 클러스터 설계상의 문제를, Daniel A. Menascé는 해석적 모형인 $M/G/1$ 큐잉[2] 모델을 사용하여 분석하였다. 그 논문에서는, 클러스터에 요청들이 포아슨 과정에 따라서 도착하고, 부하배분기는 그 클러스터의 부하를 모든 서버들 사이에 균등한 확률로 랜덤하게 배분하며, 각 웹 서버에서 하나의 요청에 대한 처리 시간은 일반적 분포를 가진다고 단순화하여, 개별 웹 서버에 대해서 익

† 홍익대학교 상경학부 교수 (교신저자)
†† 홍익대학교 상경학부 교수
논문접수: 2008년 5월 21일, 심사완료: 2008년 6월 8일
* 이 논문은 2004년도 홍익대학교 학술연구진흥비에 의하여 지원되었음.

히 알려진 $M/G/1$ 큐잉 이론을 적용함으로써 웹 사이트의 평균 반응 시간 등을 계산하였다[1].

웹 클러스터에 들어오는 요청들을 대기 상태에서 어떻게 버퍼링하며, 그 요청들을 어떤 방식으로 서버들에 배분하느냐도 그 웹 사이트의 성능에 결정적인 영향을 미친다. Lazar Kirchev는 이러한 이슈들을 시뮬레이션 접근법을 사용하여 분석하고자 시도하였다[3]. 하지만, 그 시뮬레이션 결과에서 중대한 오류를 발견할 수 있다. (이 Kirchev 논문에서 시스템 부하의 증가에 따라 반응시간이 감소하는 것으로 나와 있는 그 시뮬레이션 결과 그래프들은 분명한 오류이며, 그 논문에서 시스템 부하를 나타내고 그 그래프들에서 가로축에 해당하는 ρ에 대한 정의는 (λ/μ)가 아니라 $(\lambda/n\mu)$로 정정되어야 한다.)

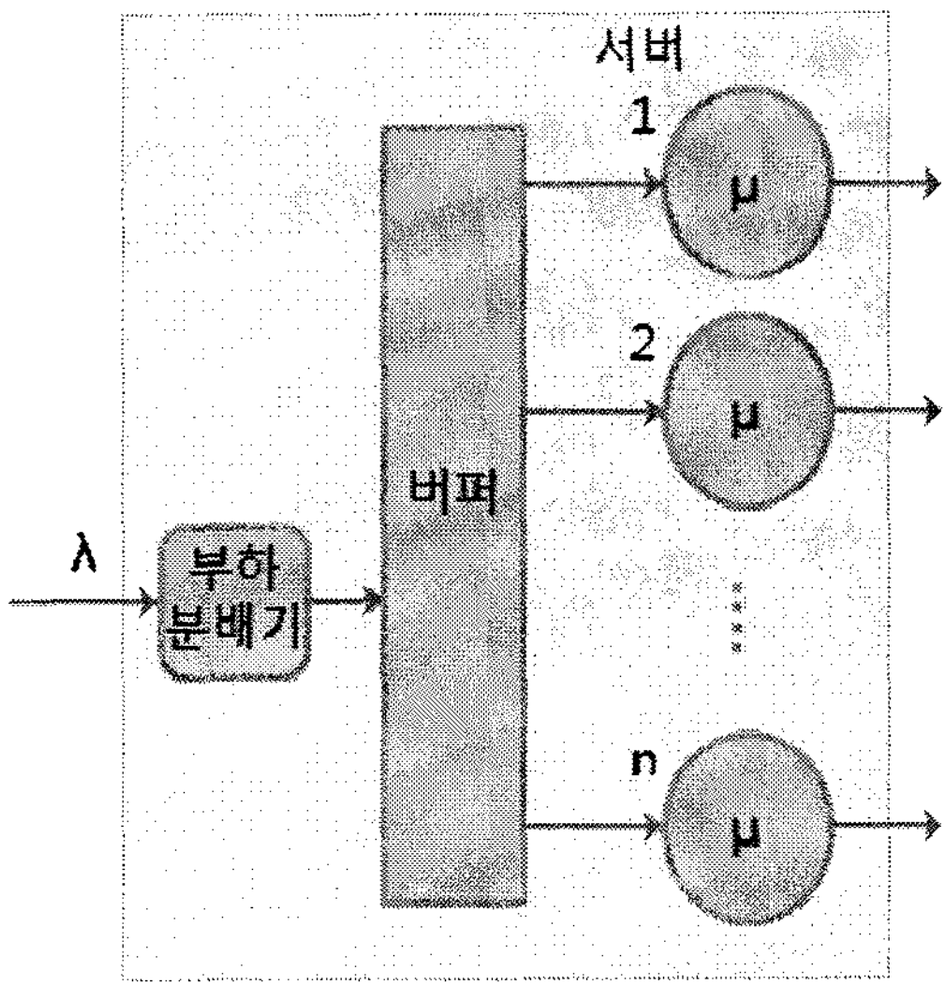
본 논문에서는 웹 클러스터 구성을 위한 여러 방안들을, 시뮬레이션 모델을 이용하여 한 요청(request)을 처리하는데 소요되는 반응 시간을 주요 성능 척도로 삼아, 비교 분석하였다. 웹 클러스터에서 버퍼링 방식, 부하 배분 정책, 개별 서버의 서비스 처리 용량에 따른 클러스터 전체 성능의 차이를 살펴보았다.

2. 웹 클러스터 구성 방안

<그림 1>과 <그림 2>에서 나타낸 것처럼, 여기에서 웹 사이트 클러스터는 n 대의 동일한 웹 서버들과 하나의 부하배분기로 이루어져 있고, 그 사이트에 입력되는 트래픽은 초당 λ 요청(requests)이며, 부하배분기가 그에 따른 부하를 모든 서버들 사이에 정해진 부하 배분 정책에 따라 적절히 할당한다고 가정한다. 그 클러스터에 n 대의 서버들은 각각 초당 μ 요청의 서비스 처리율을 가지며, 그 클러스터의 총 서비스 처리 용량은 초당 $n\mu$ 개의 요청이다. 여기서, 안정화 상태의 시스템을 연구 대상으로 하는 바, $\lambda < n\mu$ 관계가 성립되도록 한다.

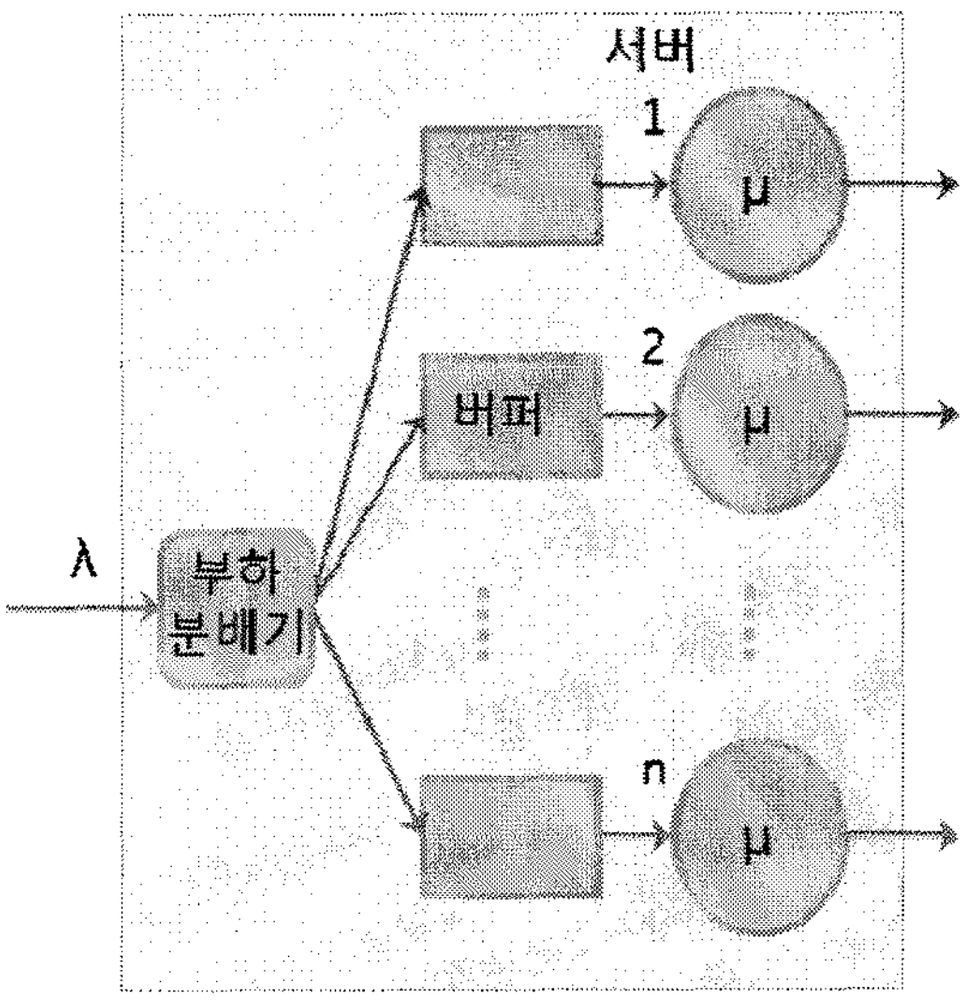
부하배분기가 한 웹 사이트에 부과되는 부하(load)를 그 클러스터에 있는 웹 서버들에게 배분하여 넘기기 바로 직전까지, 그 사이트에 들어오는 요청들을 버퍼링하는 방식도 그 사이트의 성능에 상당한 영향을 미친다. 본 논문에서는 그 버퍼링 방식으로는 공통 버퍼 (common buffer) 방식과 개별 버퍼 (local buffer) 방식을 고려하였다.

공통 버퍼 방식은, <그림 1>에서 나타낸 바와 같이, 클러스터를 구성하는 웹 서버들 전단에 부하배분기와 함께 공통 메모리 버퍼를 두어 이것을 웹 서버들 사이에 공유하도록 하며, 배분기가 그 버퍼에서 대기하는 요청들을 가용한 서버들에 나누어 주는 방식이다.



〈그림 1〉 공통 버퍼를 사용한 클러스터 구성

개별 버퍼 방식은 <그림 2>에서 보는 것처럼 각 서버가 개별적으로 국지적인 버퍼를 가지고 있다. 그 웹 사이트에 들어오는 요청들은 그 클러스터에서 적용하는 부하 배분 정책에 따라 각 웹 서버에 배정되며 그 대기 공간으로 개별 국지 버퍼를 사용한다.



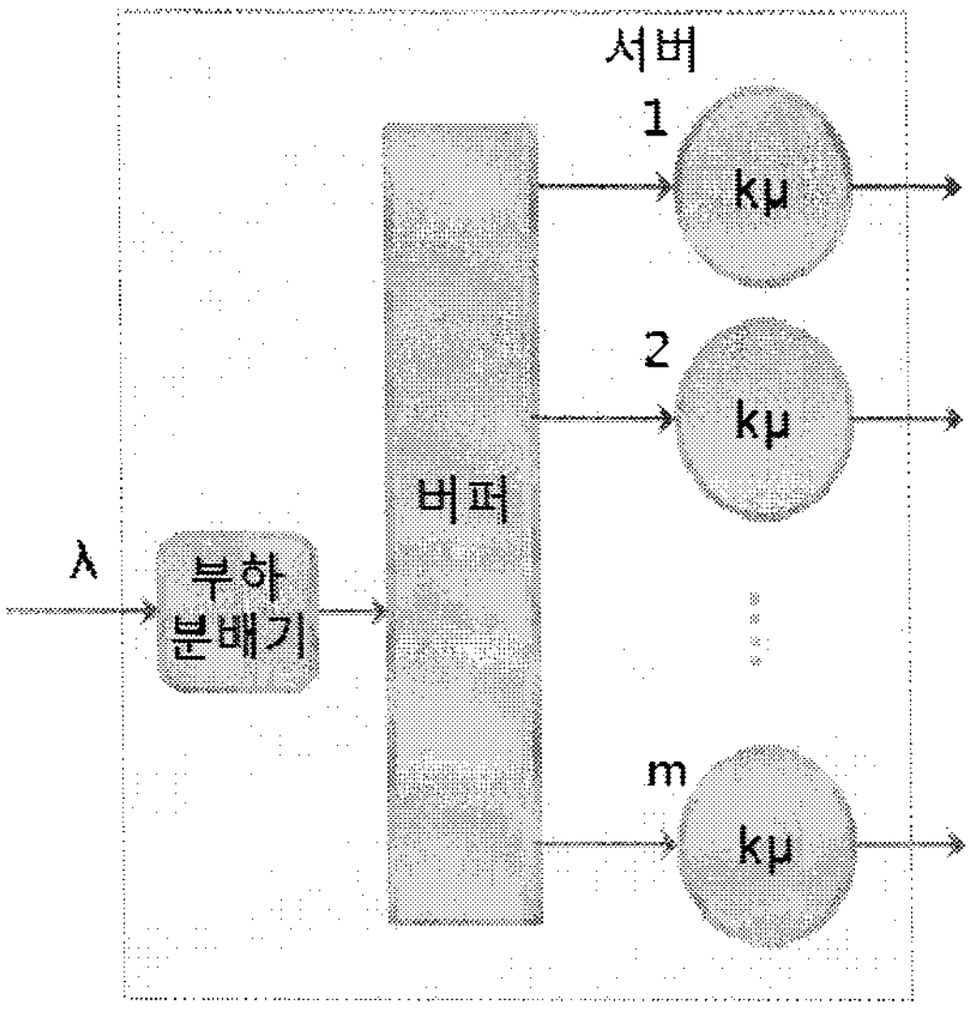
〈그림 2〉 개별 버퍼를 사용한 클러스터 구성

부하배분기가 그 사이트의 요청들을 개별 버퍼를 지닌 각 웹 서버로 배정할 때 적용하는 부하 배분 정책(load

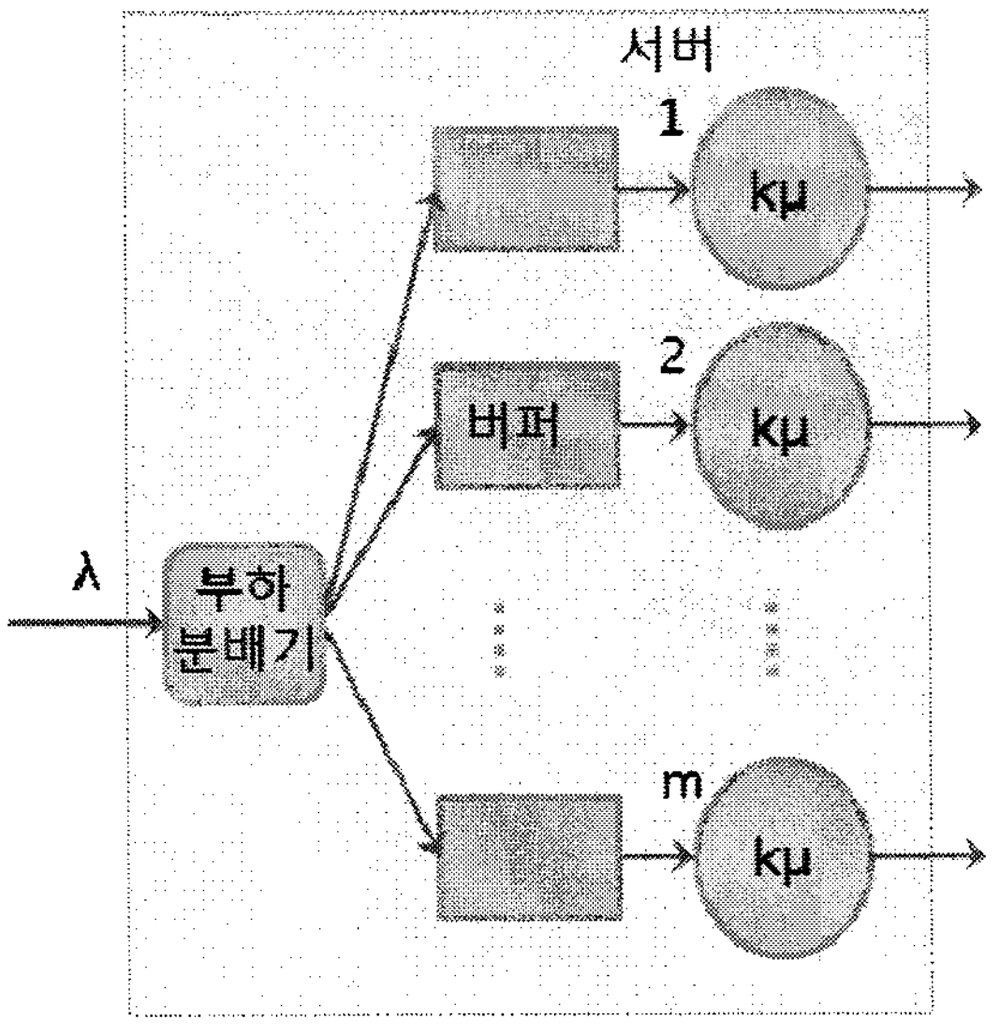
balancing policy)으로는 무작위(random) 균분 배정, 최소 큐(least queue) 서버 우선 배정, 라운드 로빈(round robin) 방식, 최저 부하(least load) 서버 우선 배정 등의 방식들을 적용할 수 있다[5, 7]. 본 논문에서는 웹 클러스터의 부하 배분 정책(load balancing policy)으로 무작위(random) 균분 배정, 라운드 로빈(round robin) 배정, 최소 큐(least queue) 서버 우선 배정의 방식을 대상으로 비교 검토하였다.

<그림 3>과 <그림 4>는 m 대의 동일한 웹 서버들로 이루어진 클러스터들을 나타내고 있다. 여기서, $m < n$ 이다. 이 클러스터들은 보다 작은 수효의 서버들로 이루어져 있지만, 그 각 서버는 보다 더 큰 처리 용량을 갖는다. 각 서버 용량은 초당 $k\mu$ 개의 요청을 처리할 수 있으며 $(k > 1)$, 그 클러스터의 총용량은 초당 $mk\mu$ 개의 요청을 처리할 수 있다. 여기서, 시스템의 안정화를 위하여, $\lambda < mk\mu$ 관계가 성립되도록 한다.

다음 절에서 시뮬레이션 모델링에 의한 수평적 증설(scaling out) 대 수직적 증설 (scaling up) 방안들에 대한 분석에는, 상호 비교의 편이를 위하여 이 두 구성안에 대한 각각의 총 처리 용량 $n\mu$과 $mk\mu$이 서로 같은 것으로 (즉, $m = n/k,\ k > 1$로) 가정한다.



〈그림 3〉 공통 버퍼, 고속 서버를 사용한 클러스터
(m과 k는 $m = n/k,\ k > 1$인 정수)



〈그림 4〉 개별 버퍼, 고속 서버를 사용한 클러스터 구성

3. 시뮬레이션 모델링 및 결과

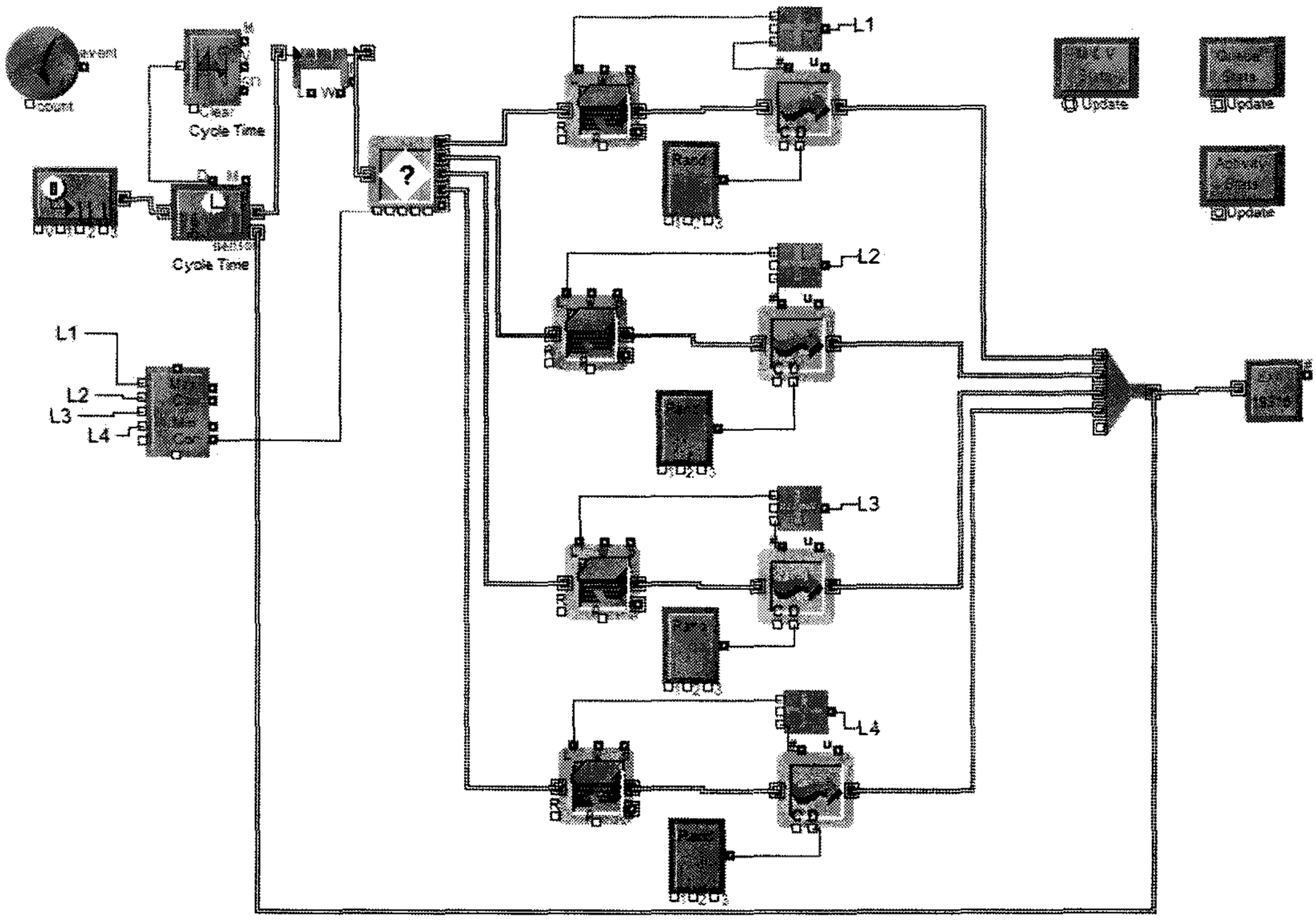
3.1 모델링

다양한 방식으로 구성되는 웹 클러스터들의 행태를 보다 정확하게 이해할 수 있는 결과들을 얻기 위해서는 상세하고 복잡한 모형들이 요구될 수 있겠지만, 비교적 간단한 시뮬레이션 모형들을 통해서도 여러 구성 방안들에 대해서 그 성능 특성을 살피고 비교분석할 수 있는 결과를 얻을 수 있다.

본 연구에서는 앞 절에서 설명한 여러 웹 클러스터 구성 방안들에 대해서 ExtendSim[3] 시뮬레이션 툴에 의한 시뮬레이션 모델들을 수립하여, 웹 클러스터에서 한 요청에 대한 평균 반응 시간 등의 성능 척도 값들을 구하여 분석하였다.

<그림 5>는 개발하여 사용한 ExtendSim 시뮬레이션 모델들의 일례로서, 웹 서버 4 대로 구성된 웹 클러스터에서 개별 버퍼 방식과 최소 큐 서버에 우선 배정의 부하 배분 정책을 사용하는 경우에 대한 것을 보여주고 있다.

클러스터들에 요청들이 포아슨 과정(Poisson process)에 따라서 도착하고, 서버에서 하나의 요청에 대한 처리 시간은 초지수 분포(hyper-exponential distribution)을 가지며, 부하배분기가 클러스터에 있는 모든 서버들 사이에 무작위 균분이나 최소 큐 서버 우선 등의 해당 부하 배분정책에 따라 그 부하를 배분하는 것으로 모델링하였다.



〈그림 5〉 ExtendSim를 사용한 시뮬레이션 모델의 일례 (개별 버퍼링, 최소 큐에 배분, $n=4$)

우리가 웹 서버에서 각 요청에 대한 서비스 시간 분포로서 초지수 분포를 사용한 것은 웹 사이트들로부터 내려 받는 파일들의 크기에서 높은 변이도를 반영할 수 있도록 한 것이다. 일반적으로 서비스 시간을 나타내는데 지수 분포나 얼랑 분포를 흔히 사용하지만, 이 분포들에 비하여 초지수 분포가 그 확률변수의 변이계수(coefficient of variation)의 값이 크다[6]. 따라서, 웹 서비스 요청에 대한 서비스 시간의 큰 변이성을 반영하여 모델링 하는데 초지수 분포가 보다 더 적합하다고 판단한다.

요구되는 성능 수준의 서비스를 제공하기 위해서 다수의 저용량 저가 서버들을 사용할 것인가 아니면 소수의 대용량 고가 서버들을 사용할 것인가 하는 문제는 웹 클러스터 구성에서 고려해야 할 전형적인 사안이다. 고용량 서버를 채용하는 scaling up 방식을 적용한 클러스터 구성안을 분석하기 위해서, 그 서비스 시간은 보통 서버의 그것과 마찬가지로 초지수 분포를 갖되 그 평균 서비스 처리율이 k배 큰 것으로 가정하였다. 이러한 분석은 <그림 3>과 같이 공통 버퍼를 사용하는 클러스터 구성안을 대상으로 하였다.

이후에 논의되는 모든 시뮬레이션 분석 결과들에 대해서, 달리 언급하지 않는다면 다음과 같은 파라메터 값들을 사용한 것이다.

$n=2,4,$ 또는 $8,\ k=2,\ \mu=20$ 요청/초

시뮬레이션에 의한 수치분석을 위한 편의상, m과 k는 $m=n/k,\ k>1$을 만족하는 정수의 값들을 사용하였다.

3.2 결과 분석

<그림 6>, <그림 7>, <그림 8>은 서버들에 각자 개별 버퍼를 두는 클러스터 구성에서 부하 배분 정책으로서 부하 무작위 균분(uniformly random), 라운드 로빈(round robin) 배정, 최소 큐(least queue) 서버에 배정의 정책를 적용하는 세 방안들과, 웹 서버들에 개별 버퍼 대신에 공통 버퍼를 두어 운영하는 방안, 그리고 보통 서버 대신에 고속의 서버를 사용하여 구성하는 방안의 성능 차이를 쉽게 비교할 수 있도록 나타낸 것이다. 클러스터에 웹 서버의 수 $n=2,4,8$ 각각에 대해서, 서버 활용도 $(\lambda/n\mu)$의 함수로서 한 요청에 대한 평균 반응 시간의 값들을 그레프로 보여주고 있다. 그 차트들의 범례 설명에서 UR은 부하 무작위 균분, RR은 라운드 로빈 배

정, LQ는 최소 큐 서버에 배정 정책을 나타낸다.

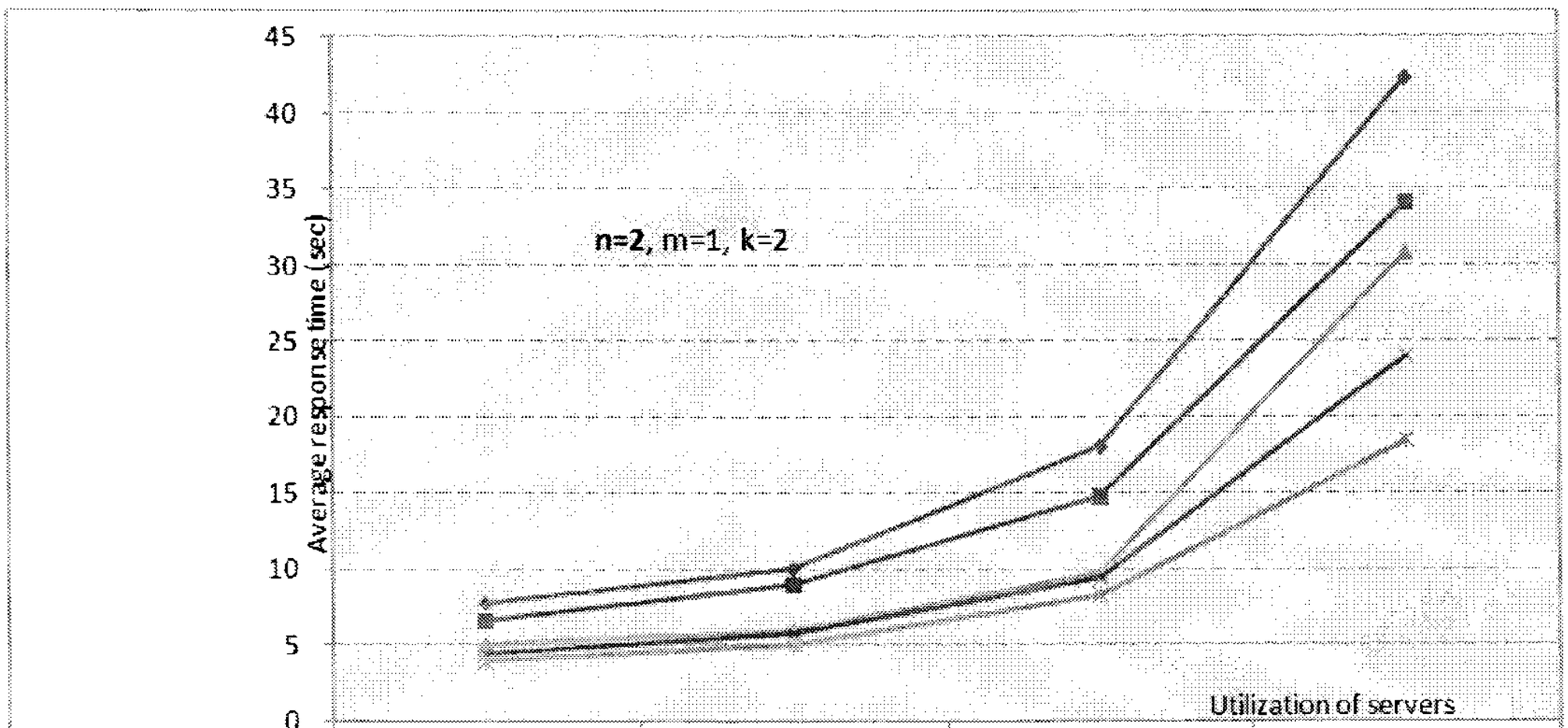
이 시뮬레이션 결과에서 클러스터 내에 웹 서버들의 공통 버퍼링이 개별 버퍼링에 비하여 성능 면에서 우월하다는 것을 알 수 있다. 웹 서버들 간에 부하를 배분하는 방식으로서는, 라운드 로빈(RR)이 무작위 균분(UR)보다 더 좋은 성능을 얻을 수 있게 하지만, 최소 큐(LQ) 서버에 배정 정책이 라운드 로빈 정책 보다 더 우수하다는 것을 확인할 수 있다. 그리고, 클러스터 전체의 서비스 처리 용량이 서로 같은 경우라도, 서비스 율이 큰 서버들로 구성된 웹 클러스터가 서비스 용량이 상대적으로 작은 서버들로 구성된 클러스터 보다 성능에서 유리하다는 것을 알 수 있다. 이러한 방안들 간의 상대적 우열은 시스템 부하 또는 서버 활용도가 증가할수록 더 커지는 경향을 보여주고 있다.

<그림 9>에서도 <그림 6, 7, 8>과 유사하게 버퍼링 방식, 부하배분 정책, 서버 용량에 따른 성능을 비교하여 볼 수 있다. 하지만, 이 그래프는 서비스 시간이 얼랑(Erlang) 분포를 따른다고 가정하고 $n=8$, $k=2$, $m=4$인 경우에 대하여 얻은 결과이다. 이것을 서비스 시간의 분포로 초지수 분포를 적용하고 $n=8$에 대하여 얻은 결과인 <그림 8>과 비교하여 보면, 서비스 시간의 평균은 같은 경우라도 웹 서버에서 내려 받는 파일 크기와 그 처리 시간의 변이도가 크면 평균 반응 시간도 증가한다는 것을 알 수 있다.

<그림 10>, <그림 11>, <그림 12>는 웹 서버들에 각자 개별 버퍼를 두는 클러스터 구성에서, 서버 수 $n=2,4,8$일 경우에 대하여 클러스터에 서버 활용도 $(\lambda/n\mu)$의 함수로서 한 요청에 대한 평균 반응 시간의 값들을 그래프로 보여주고 있다. 그 부하배분 정책으로서 부하 무작위(uniformly random) 균분, 라운드 로빈(round robin) 배정, 최소 큐(least queue) 서버에 배정의 정책을 적용한 각각의 결과이다.

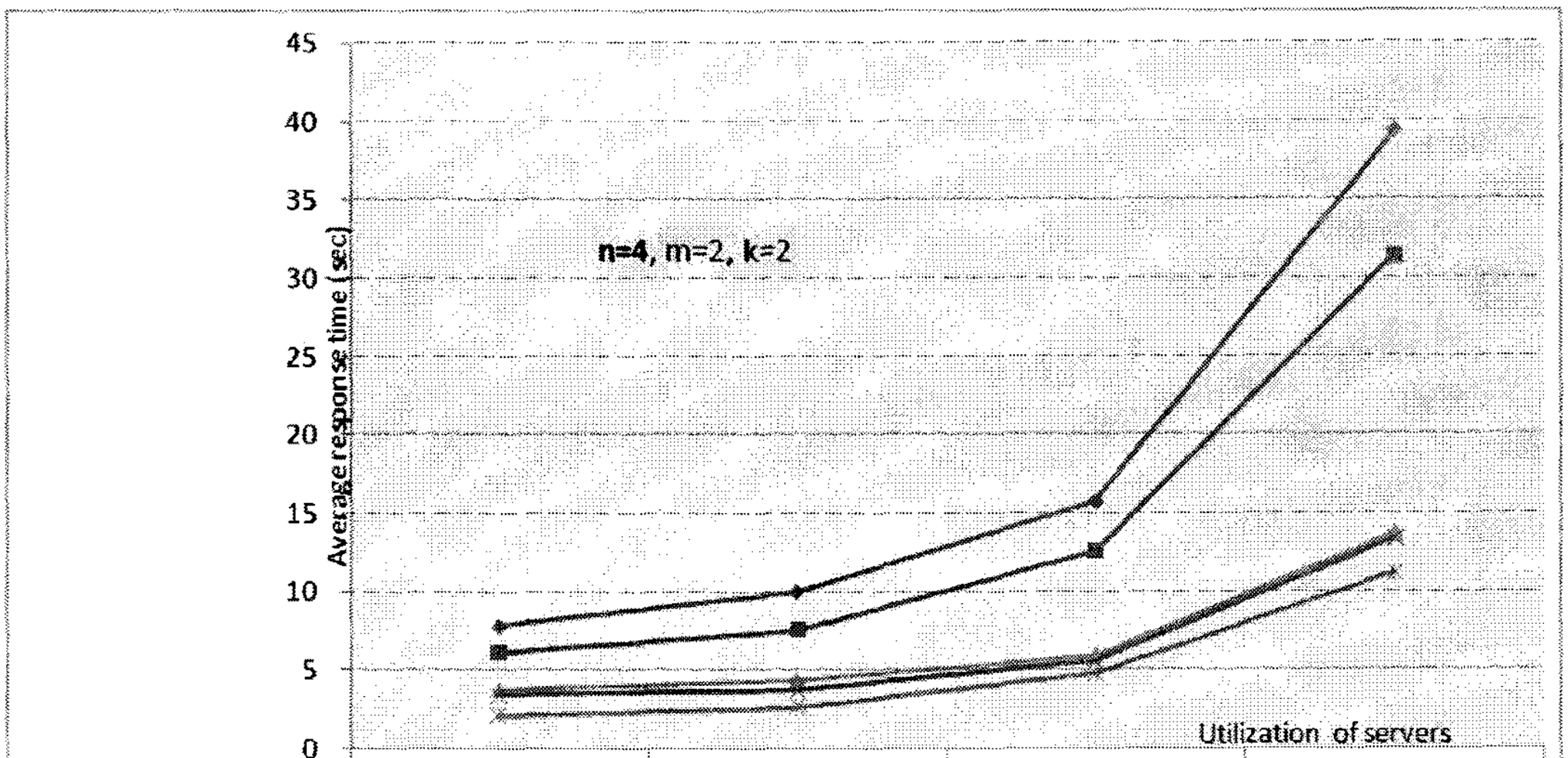
<그림 13>은 클러스터 내에 웹 서버들에 공통 버퍼를 두는 구성에서, 서버 수 $n=2,4,8$일 경우에 대하여 클러스터에 서버 활용도 $(\lambda/n\mu)$의 함수로서 한 요청에 대한 평균 반응 시간의 값들을 그래프로 보여주고 있다.

<그림 14>는 클러스터 용량 확장 방법으로 수평적 증설(scaling out)과 수직적 증설(scaling up)를 비교할 수 있도록 한 것이다. 서버의 서비스 처리율이 $k=2$로 두 배 차이 나고 $n=2,4,8$이고 $m=1,2,4$인 경우에 대해서, 각 클러스터에 서버 활용도 $(\lambda/n\mu)$ 또는 $(\lambda/mk\mu)$의 함수로서 한 요청에 대한 평균 반응 시간의 변화를 보여주고 있다. 이 경우에 서버 활용도는 두 클러스터들에 있어서 같다.



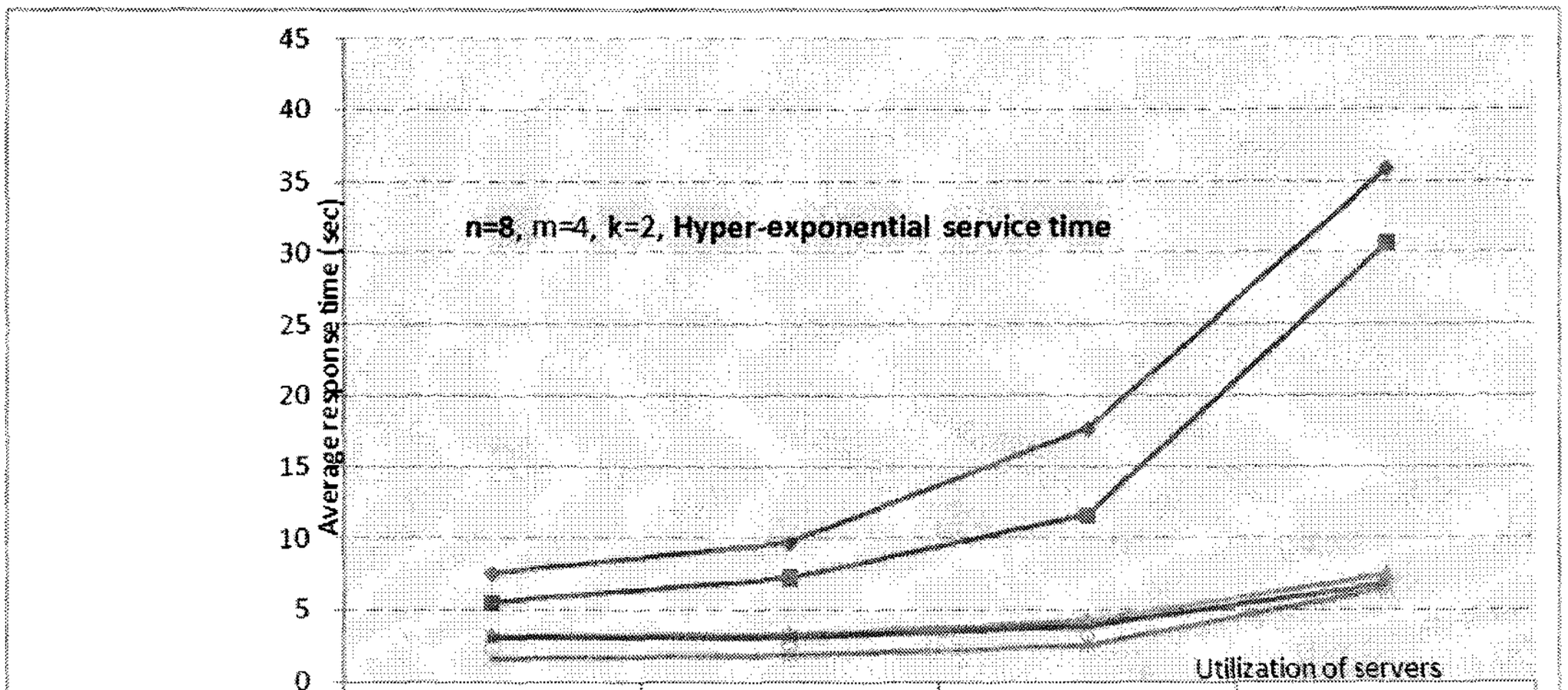
	0.5	0.6	0.75	0.9
Local Buffers, UR	7.71	9.99	18.01	42.28
Local Buffers, RR	6.56	8.97	14.76	34.09
Local Buffers, LQ	5.04	6.00	9.79	30.84
Common Buffer	4.40	5.72	9.42	24.02
Fast Servers with CB	3.98	5.01	8.31	18.38

〈그림 6〉 버퍼링 방식, 부하배분 정책, 서버 용량에 따른 성능 비교 ($n=2$)



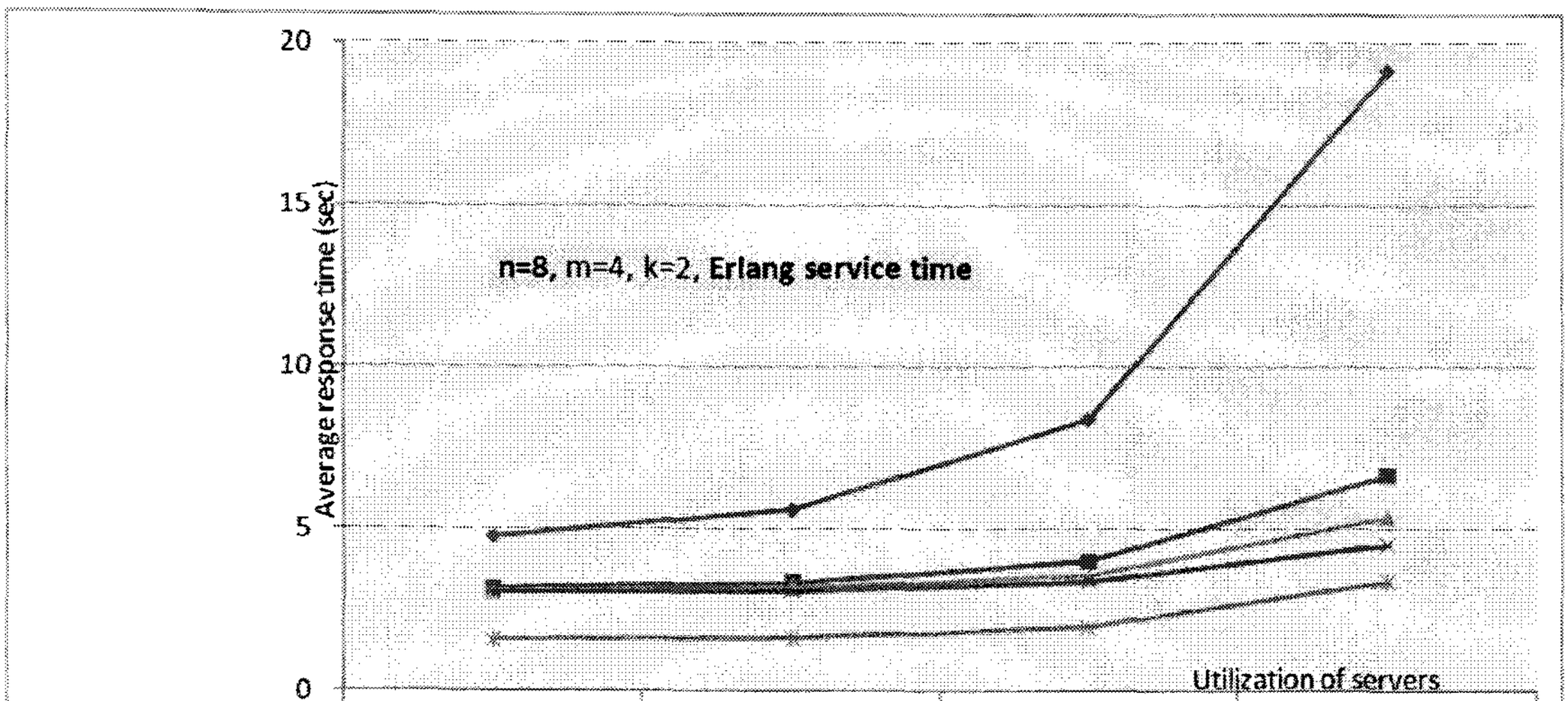
	0.5	0.6	0.75	0.9
Local Buffers, UR	7.80	10.00	15.76	39.39
Local Buffers, RR	6.11	7.53	12.61	31.42
Local Buffers, LQ	3.76	4.37	5.93	13.70
Common Buffer	3.37	3.70	5.52	13.36
Fast Servers with CB	2.12	2.67	4.77	11.17

〈그림 7〉 버퍼링 방식, 부하배분 정책, 서버 용량에 따른 성능 비교 ($n=4$)



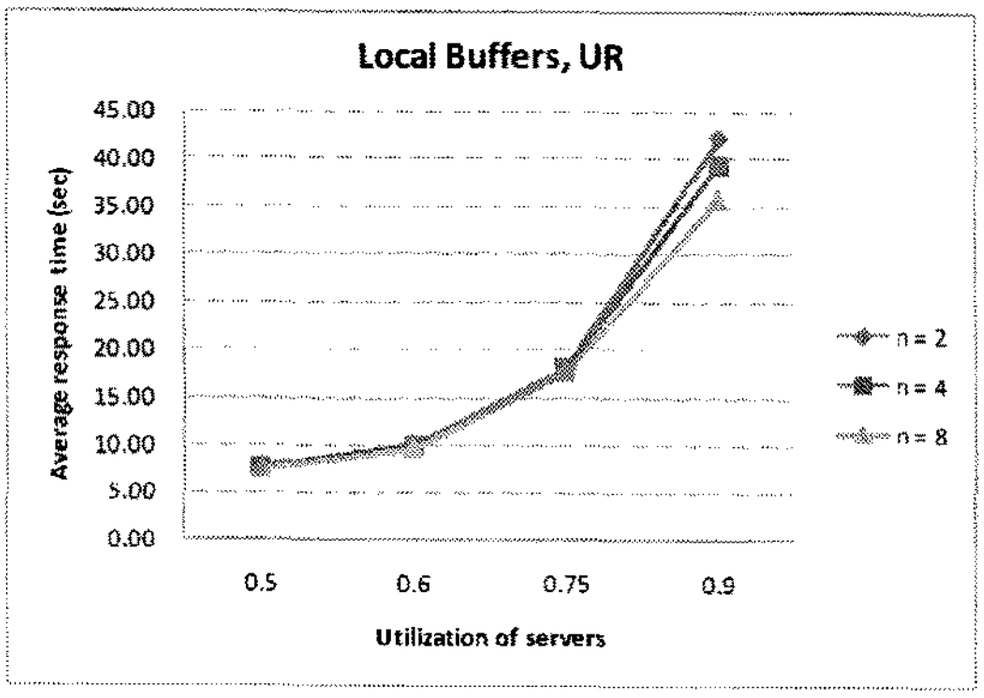
	0.5	0.6	0.75	0.9
Local Buffers, UR	7.52	9.61	17.61	35.83
Local Buffers, RR	5.51	7.21	11.53	30.61
Local Buffers, LQ	3.23	3.36	4.21	7.45
Common Buffer	3.09	3.11	3.83	6.73
Fast Servers with CB	1.67	1.87	2.61	6.46

〈그림 8〉 버퍼링 방식, 부하배분 정책, 서버 용량에 따른 성능 비교 ($n=8$)

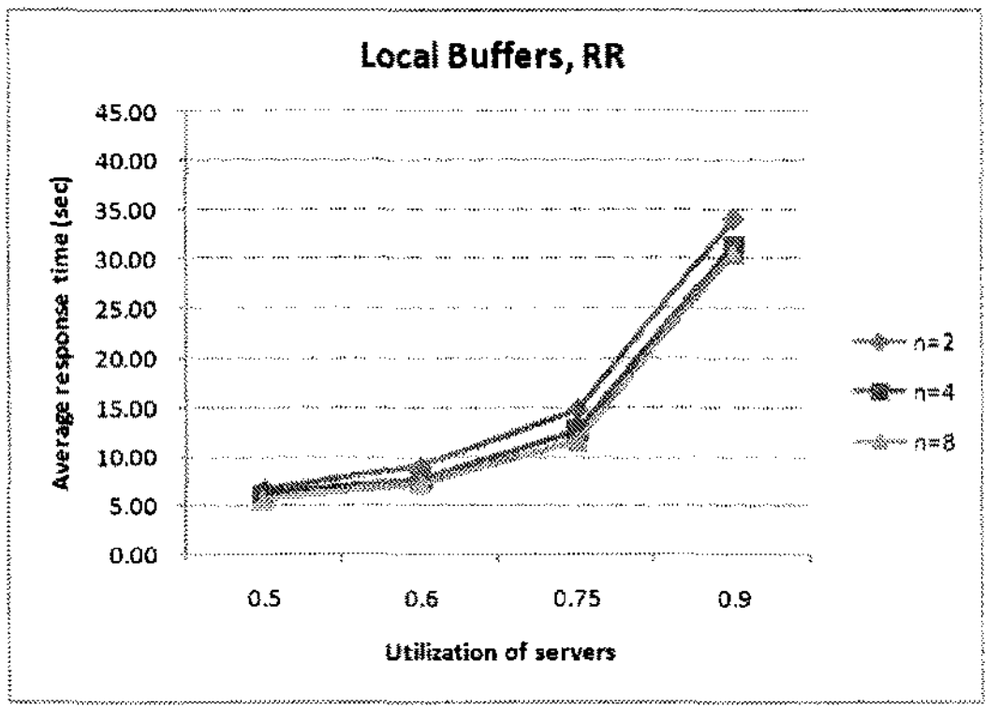


	0.5	0.6	0.75	0.9
Local Buffers, UR	4.77	5.61	8.41	19.14
Local Buffers, RR	3.17	3.34	4.01	6.65
Local Buffers, LQ	3.08	3.18	3.56	5.37
Common Buffer	3.06	3.08	3.38	4.53
Fast Servers with CB	1.60	1.67	1.98	3.39

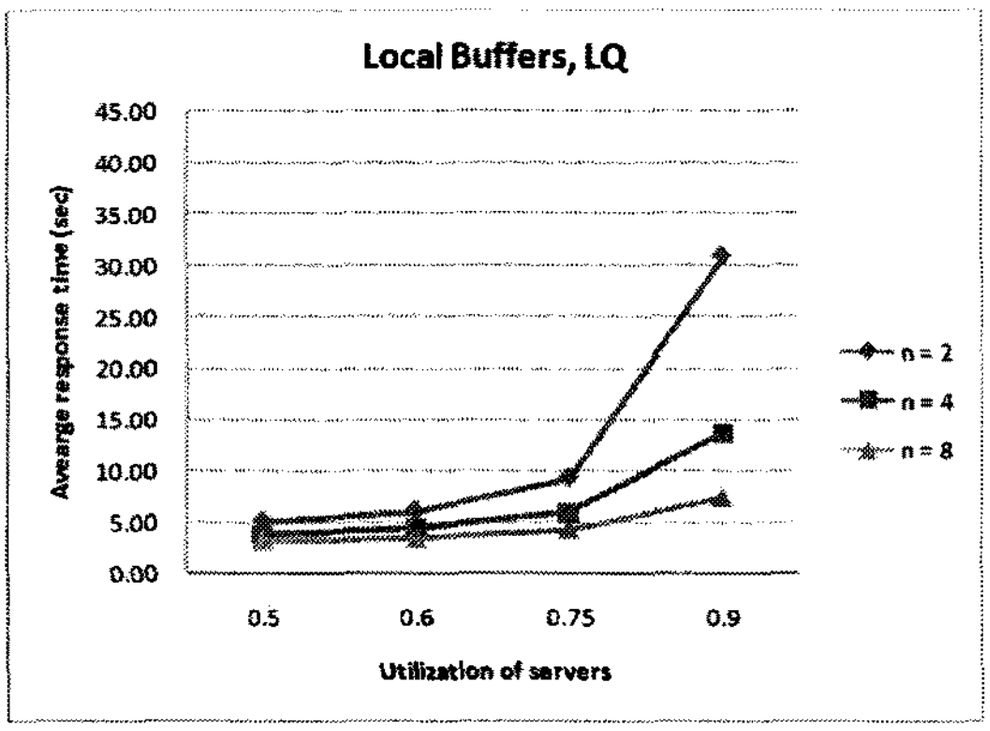
〈그림 9〉 버퍼링 방식, 부하배분 정책, 서버 용량에 따른 성능 비교
($n=8,\ k=2,\ m=4$, 서비스 시간 분포로서 얼랑 분포 적용시)



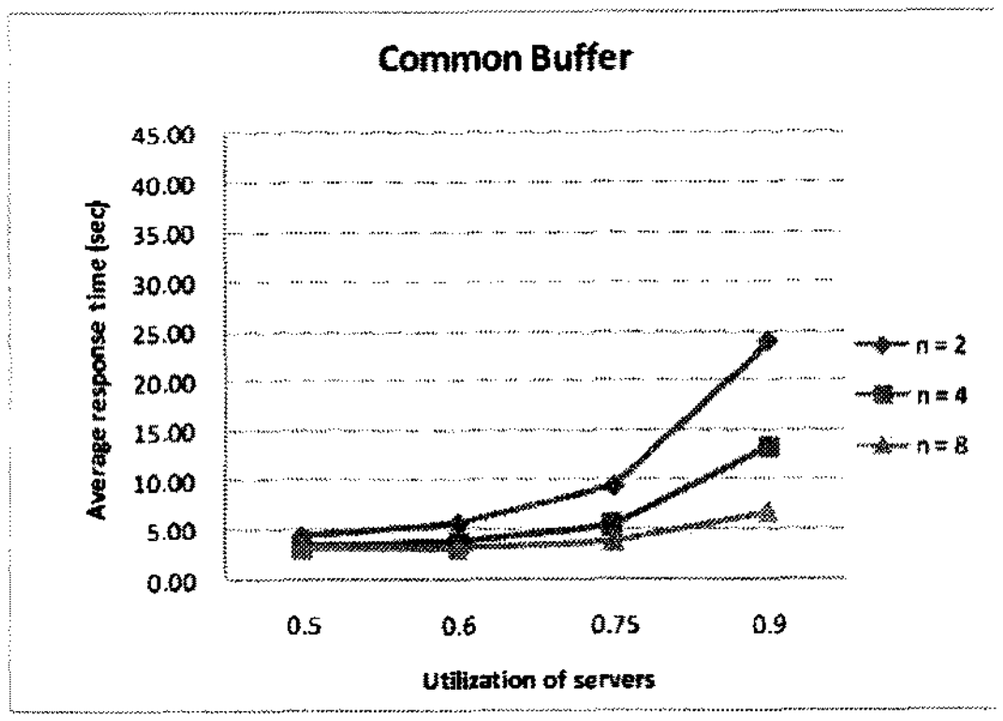
〈그림 10〉 개별 버퍼, 무작위 균분 결과



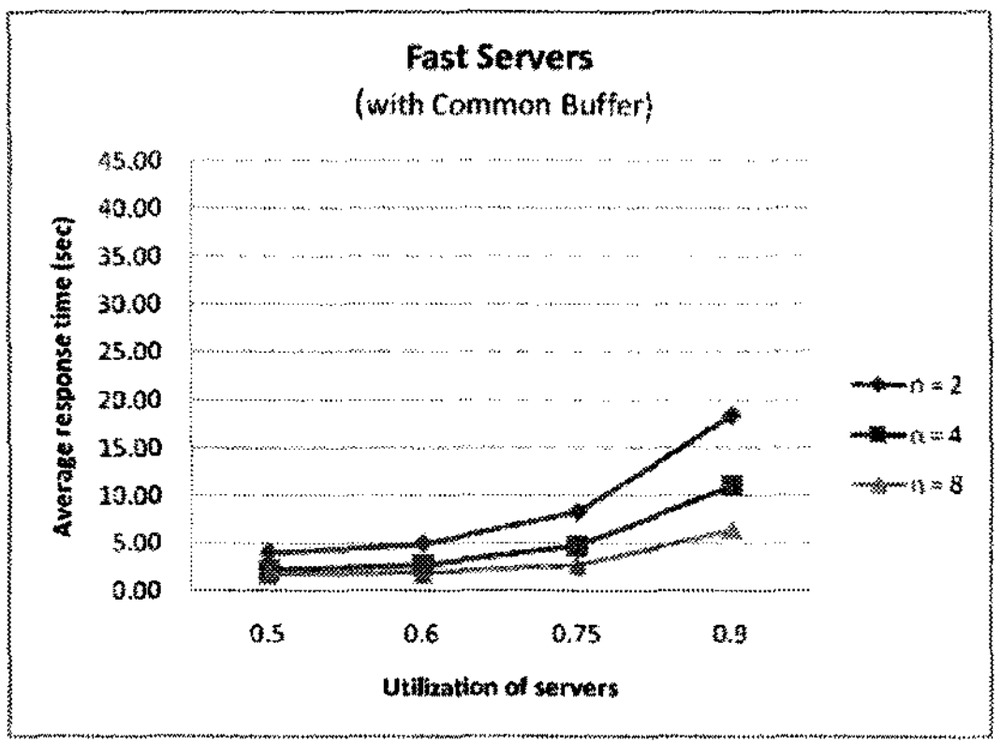
〈그림 11〉 개별 버퍼, 라운드 로빈 배분 결과



〈그림 12〉 개별 버퍼, 최소 큐의 서버에 배정 결과



〈그림 13〉 공통 버퍼 적용 결과



〈그림 14〉 클러스터 전체 서비스율은 다른 경우와 같은 조건 하에, 고속 서버 사용 결과

4. 결 론

대규모 웹 사이트는 흔히 다수 웹 서버들의 클러스터로서 구성한다. 이러한 방법을 사용하여 여러 컴퓨터들 간에 처리 부하를 분산시키고 반응 시간을 줄이며, 그 사이트의 가용도와 확장성을 증가시킨다.

본 논문에서는 그러한 웹 클러스터의 구성에서 웹 사이트 요청들에 대한 버퍼링 방식, 부하 배분 정책, 서버 용량의 측면들에서 여러 가능한 방안들을 설정하고 그에 대한 성능을 시뮬레이션 모델을 통해서 살펴 보았다. 그 결과로 웹 클러스터에서 요청들의 대기 공간으로 웹 서버들을 위한 공통 버퍼를 두는 것이 서버 마다 국지적인 개별 버퍼를 두는 것 보다 성능에서 우월하며, 클러스터에서 웹 서버들에 부하 배분 정책으로서 최소 큐에 배정하는 방식이 라운드 로빈이나 무작위 균분 방식에 비하여 보다 우수하다는 것을 확인할 수 있었다.

그리고, 서비스율이 높은 고속의 웹 서버들로 이루어진 클러스터가 상대적으로 서비스율이 낮은 저속의 웹 서버들 다수로 이루어진 클러스터에 비하여, 클러스터의 총 서비스율은 서로 같을지라도, 그 성능 면에서 우월하다는 것을 살필 수 있었다. 따라서, 웹 서버의 비용이 그 처리 용량에 정비례한다거나 서버 용량이 증가함에 따라 규모의 경제가 있어 단위 용량 당 비용이 감소한다면, 비용 대비 성능의 측면에서 고속 웹 서버들로 구성된 클러스터가 유리하다. 하지만, 단위 용량 당 서버 비용이 용량에 따라 증가하는 식으로 서버의 비용이 그 서비스 처리 용량의 상향비선형(superlinear) 함수의 형태로 늘어나는 경우이라면, 비용 대비 성능 향상의 관점에서 웹 서버의 용량에 대한 절충점을 모색해야 될 것이다. 또한, 고속이나 저속의 웹 서버의 개별 신뢰도가 서로 비슷한 조건이라면, 웹 사이트의 가용도 측면에서는 저속이지만 보다 다수의 웹 서버들로 구성된 클러스터가 보다 바람직하다.

참 고 문 헌

[1] Daniel A. Menascé, Trade-offs in Designing Web Clusters, IEEE INTERNET COMPUTING, SEPTEMBER · OCTOBER 2002, pp. 76-80

[2] Donald Gross, Carl M. Harris, *Fundamentals of Queueing Theory, Second Edition*, John Wiley & Sons, 1985

[3] Lazar Kirchev, Simulation Modeling of Replicated Clustered Web Services, FIRST INTERNATIONAL CONFERENCE on INFORMATION SYSTEMS & DATAGRIDS, 17-18 February 2005, Sofia, Bulgaria

[4] Manuel Laguna, Johan Marklund, *Business Process Modeling, Simulation, and Design*, Pearson Prentice Hall, 2005

[5] Valeria Cardellini, Michele Colajanni, and Philip S. Yu, "Dynamic Load Balancing on Web Server Systems," IEEE Internet Computing, May/June 1999, pp. 28-39.

[6] Wikipedia, Hyper-exponential distribution, http://en.wikipedia.org/wiki/Hyper-exponential_distribution

[7] Yong Meng Teo, Rassul Ayani, Comparison of Load Balancing Strategies on Cluster-based Web Servers, SIMULATION, November-December 2001, pp. 185-195

강 성 열

1981 서울대학교 산업공학과 (학사)
1983 서울대학교 산업공학과 (석사. 경영과학 전공)
1992 미국 Georgia Tech, 산업시스템공학과 (박사. Stochastic Modeling, 정보통신망 전공)
1983~1998 ETRI 책임연구원
1999~2000 KAIST 테크노경영대학원 초빙교수
2000~현재 홍익대학교 상경학부 교수
관심분야: 정보통신, 인터넷, Stochastic Modeling
E-Mail: skang@hongik.ac.kr

송 영 효

1980 서울대학교 산업공학과 (공학사)
1982 KAIST 경영과학과 (이학석사)
1992 미국 RPI대학교 Lally경영대학원 경영학과(경영학 박사)
1992~1993 삼성데이터시스템 선임
1993~1996 삼성증권 상품개발팀장
1996~현재 홍익대학교 상경대학 교수
관심분야: 시뮬레이션, 데이터마이닝
E-Mail: songyh@wow.hongik.ac.kr